"Express Mail" mailing label number: EV 164034115US

Date of Deposit: November 18, 2003

Attorney Docket No. 15141US02

STORING MACROBLOCKS FOR CONCATENATED FRAMES

RELATED APPLICATIONS

[0001] This application claims priority to Provisional Application Serial No. 60/495,405, Attorney Docket No. 15141US01, filed August 15, 2003, entitled "Storing Macroblocks for Concatenated Frames".

[0002] This application is related to Provisional App. Ser. No. 60/484,512, Attorney Docket No. 15030US01, filed July 2, 2003, entitled "System, Method, and Apparatus for Efficiently Storing Macroblocks in SD-RAM", and Provisional App. Ser. No. 60/484,830, Attorney Docket No. 15054US01, filed July 3, 2003, entitled "SYSTEM, METHOD, AND APPARATUS FOR EFFICIENTLY STORING MACROBLOCKS", each of which are incorporated by reference for all purposes.

FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

[0003] [Not Applicable]

[MICROFICHE/COPYRIGHT REFERENCE]

[0004] [Not Applicable]

BACKGROUND OF THE INVENTION

[0005] One of the major challenges in designing a memory subsystem is organizing data in such a way, as to enable efficient memory access that would increase the system throughput. Data organization in the memory becomes even more a bigger challenge in a UMA (Unified Memory Architecture) subsystem, where it has a direct impact on the efficiency of the system as a whole. Therefore data in the memory should be organized in such a way that, a high bandwidth client (a client is an agent that initiates data transfer between itself and the memory subsystem) benefits the most without compromising the access efficiency of the other low bandwidth clients. In other words the data organization in the memory should help reduce the DDR-SDRAM overheads for high bandwidth clients, which in turn would improve the efficiency of the memory subsystem as a whole.

[0006] In a video decompression-engine (a.k.a video decoder) a substantial portion of the system bandwidth is utilized in transacting video pixel data. The video decoder uses the neighboring macro-block (a macro block is a 16x16 pixel block) data from the previous and future frames of the video to predict the current macro-block information. Thus the right choice would be to have a memory subsystem that is macro-block oriented.

[0007] However the current column sizes in the DDR-SDRAM technology does not allow us to pack a full macro-block row information in one bank of the DRAM for a SD size picture. At the same time, a very simple linear arrangement of macro-block continuously in the same bank of the DDR-SDRAM would increase the SDRAM overheads, as an adjacent or vertical neighbor macro-block fetch would require a different row of the same bank to be activated. In such a case, the current row of the current bank needs to be precharged and a new row of the same bank needs to be activated, thus resulting in roughly 6-clocks overhead per row change. In the worst case, a particular video decode fetch could involve four macro-blocks worth data, lying in four different rows of the same bank, resulting in as high as 18 (three row switching) clocks overhead. On the other hand, if the adjacent or vertical macro-block were to exist in different banks of the DRAM, it would be possible to reduce the SDRAM overhead to zero in the best case and the worst case numbers will be much less that 18 clocks.

[0008] Conventionally, four macro blocks worth data are packed into one bank, before switching to the next bank of the DDR-SDRAM. This packing would be efficient for images, whose number of horizontal macro-block (NMBX) follows the equation,

NMBX = 16*N + 8

(where N is any positive integer)

[0009] The above equation ensures efficient data fetching & packing for a HD size picture (NMBX = 120). However for a SD size picture (NMBX = 45), the closest value of N, that satisfies the above equation = 4, resulting in NMBX required = 56. This means we have 11 macro-blocks, wasted for every macro-block row of the image. For a SD size picture this would be roughly 75Kbytes of wasted memory per frame storage (roughly 20% wastage per frame).

[0010] Further limitations and disadvantages of conventional and traditional approaches will become apparent to one of skill in the art, through comparison of such systems with the present invention as set forth in the remainder of the present application with reference to the drawings.

BRIEF SUMMARY OF THE INVENTION

[0011] Presented herein is a system for storing macroblocks for concatenated frames. In one embodiment, there is a circuit for decoding video data. The circuit comprises an instruction memory and a processor. The instruction memory stores a plurality of executable instructions. The processor executes the plurality of executable instructions. The execution of the plurality of executable instructions causes storing a portion of a first frame in a row of memory, and storing a portion of a second frame in the row of memory.

[0012] In another embodiment, there is a circuit for decoding video data. The circuit comprises an instruction memory and a processor. The instruction memory stores a plurality of executable instructions. The processor executes the plurality of executable instructions. The execution of the plurality of executable instructions causes storing a first macroblock row of a first frame in a first one or more rows of memory, storing a first macroblock row of a second frame in a second one or more rows of memory, and a particular one of the first one or more rows of memory being adjacent to a particular one of the second one or more rows of memory.

[0013] In another embodiment, there is presented a method for decoding video data. The method comprises storing a portion of a first frame in a row of memory, and storing a portion of a second frame in the row of memory.

[0014] In another embodiment, there is presented a method for decoding video data. The method comprises storing a first macroblock row of a first frame in a first one or more rows of memory, storing a first macroblock row of a second frame in a second one or more rows of memory, and a particular one of the first one or more rows of memory being adjacent to a particular one of the second one or more rows of memory.

[0015] These and other advantages and novel features of the present invention, as well as details of an illustrated embodiment thereof, will be more fully understood from the following description and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0016] **FIGURE 1** is a block diagram describing the encoding of video data;

[0017] **FIGURE 2** is a block diagram of a decoder system in accordance with an embodiment of the present invention;

[0018] **FIGURE 3** is a block diagram of an exemplary reference frame;

[0019] **FIGURE 4** is a block diagram of an exemplary DRAM;

[0020] **FIGURE 5** is a block diagram of an exemplary SDTV frame;

[0021] **FIGURE 6** is a block diagram describing the storage of a frame in accordance with an embodiment of the present invention;

[0022] **FIGURE 7** is a block diagram of three horizontally concatenated frames; and

[0023] **FIGURE 8** is a block diagram describing the storage of the frames in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

[0024] Referring now to **FIGURE 1**, there is illustrated a block diagram describing MPEG formatting of video data 305. The video data 305 comprises a series of frames 310. Each frame comprises two dimensional grids of luminance Y, chroma red Cr, and chroma blue Cb pixels 315. The two-dimensional grids are divided into 8x8 blocks 335, where four blocks 335 of luminance pixels Y are associated with a block 335 of chroma red Cr, and a block 335 of chroma blue Cb pixels. The four blocks of luminance pixels Y, the block of chroma red Cr, and the chroma blue Cb form a data structure known as a macroblock 337. The macroblock 337 also includes additional parameters, including motion vectors.

[0025] The data in the macroblocks 337 is compressed in accordance with algorithms that take advantage of temporal and spatial redundancies. For example, in a motion picture, neighboring frames 310 usually have many similarities. Motion between frames increases differences between frames. Motion compensation can be used to reduce these differences. When an object moves across a screen, the object may appear in different positions in different frames, but the object does not change substantially in appearance. The picture differences can be reduced by measuring and recording the motion as a vector. The vector can be used during decoding to shift a macroblock 337 of one frame to a more appropriate part of another frame.

[0026] Accordingly, most of the macroblocks 337 are compared to portions of other frames 310 (reference frames). When an appropriate portion of a reference frame

310 is found, the differences between the portion of the other frame 310 and the macroblock 337 are encoded. The location of the portion in the reference frame 310 is recorded as a motion vector. The encoded difference and the motion vector form part of the data structure encoding the macroblock 337. In MPEG-2, the macroblocks 337 from one frame 310 (a predicted frame) are limited to prediction from portions of no more than two reference frames 310. It is noted that frames 310 used as a reference frame 310 for a predicted frame 310 can be a predicted frame 310 from another reference frame 310.

[0027] The macroblocks 337 representing a frame are grouped into different slice groups 340. The slice group 340 includes the macroblocks 337 in the slice group 340, as well as additional parameters describing the slice group. Each of the slice groups 340 forming the frame form the data portion of a picture structure 345. The picture 345 includes the slice groups 340 as well as additional parameters. The pictures are then grouped together as a group of pictures 350. The group of pictures 350 also includes additional parameters. Groups of pictures 350 are then stored, forming what is known as a video elementary stream 355. The video elementary stream 355 is then packetized to form a packetized elementary sequence 360. Each packet is then associated with a transport header 365a, forming what are known as transport packets 365b.

[0028] The transport packets 365b can be multiplexed with other transport packets 365b carrying other content, such as another video elementary stream 355 or an audio elementary stream. The multiplexed transport packets from what is known as a transport stream. The transport stream is transmitted over a communication medium for decoding and presentation.

[0029] Referring now to **FIGURE 2**, there is illustrated a block diagram of an exemplary decoder system for decoding compressed video data, configured in accordance with an embodiment of the present invention. A processor, that may include a CPU 490, reads transport stream 465 into a transport stream buffer 432 within an SDRAM 430.

[0030] The data is output from the transport stream buffer 432 and is then passed to a data transport processor 435. The data transport processor 435 then demultiplexes the transport stream 465 into constituent transport streams. The constituent packetized elementary stream can include for example, video transport streams, and audio transport streams. The data transport processor 435 passes an audio transport stream to an audio decoder 460 and a video transport stream to a video transport processor 40.

[0031] The video transport processor 440 converts the video transport stream into a video elementary stream and provides the video elementary stream to a video decoder 445. The video decoder 445 decodes the video elementary stream, resulting in a sequence of decoded video frames. The decoding can include decompressing the video elementary stream. It is noted that there are various standards for compressing the amount of data required for transportation and storage of video data, such as MPEG-2. The video decoder 445 decompresses the video data.

[0032] A display engine 450 is responsible for and operable to select a frame (or field) for display at every vertical synchronization pulse, scale the frame, render the graphics, construct the complete display, and rasterize the frame, among other functions. The rasterized frame is passed to a video encoder 455 where it is converted to analog video using an internal digital to analog converter (DAC). The digital audio is converted to analog in the audio digital to analog converter (DAC) 465.

[0033] The decoded video data includes a series of frames 310. The frames 310 are stored in frame buffers 452. The frame buffers 452 can be dynamic random access memory (DRAM) comprising 128 bit/16 byte gigantic words (gwords). As noted above, most macroblocks 337 in predicted frames 310 are encoded as an offset or difference from portions of reference frames 310. Accordingly, the MPEG video decoder 445 decodes the reference frames 310 prior to decoding predicted frames 310 that are predicted thereon. The decoded reference frames 310 are stored in frame buffers 452. The MPEG video decoder 445 fetches the portions of the reference frames 310 from the frame buffers 452 to decode macroblocks 337 from the predicted frame 310.

[0034] Referring now to **FIGURE 3**, there is illustrated a block diagram of a reference frame 310. As noted above, the frame 310 is decoded on a macroblock by macroblock 337 basis. Macroblocks 337 of another frame 310 can be encoded as an offset or difference from portions 310p of the frame 310. The portions 310p are not necessarily aligned with macroblocks 337. A portion 310p can potentially straddle four adjacent macroblocks 337. The MPEG video decoder 445 retrieves the macroblocks 337, containing the portion 310 for decoding a macroblock 337 in a predicted frame 310.

[0035] Referring now to **FIGURE 4**, there is illustrated a block diagram of an exemplary DRAM 500. The DRAM 500 comprises four banks, bank 0, bank 1, bank 2, and bank 3. Each bank comprises any number of rows 505(0)…505(n). Each row of a bank has 32 byte jumbo words (jword). The luma Y portion of the macroblocks 337 occupy 8 j-words. Therefore, each row of a bank can store the luma pixels from four macroblocks 337.

[0036] To access data, a bank is charged while the bank is charged, data from one row of the bank can be accessed. Access to other rows of the bank occurs after the first access is completed. Each memory access is associated with overhead time. As noted above, the video decoder 445 to fetch a portion for decoding a macroblock 337, the video decoder 445 may fetch up to four adjacent macroblocks. The time for decoding can be reduced by fetching each of the macroblocks 337 in pipeline. However, if the macroblocks 337 occupy different rows of the same bank, the requests cannot occur in pipeline with many internal banks of DRAMS. To avoid this, the frame 310 can be stored in a frame buffer 452 in a manner such every set of four covering macroblocks 337 are stored in either different banks or the same row of a bank.

[0037] Referring now to **FIGURE 5**, there is illustrated a block diagram of an exemplary decoded standard definition TV (SDTV) reference frame 310. The reference frame 310 can be divided into macroblocks, A0…A44, B0…B44, C0…C44, D0…D44, E0…E44, F0…F44, etc. The macroblocks comprise 16x16 blocks of luma pixels and two 8x8 blocks of chroma pixels. An SDTV reference frame comprises 45 macroblocks, e.g., A0 … A44, across each row.

[0038] Referring now to **FIGURE 6**, there is illustrated a block diagram of an exemplary frame buffer storing a frame in accordance with an embodiment of the present invention. The frame buffer comprises four banks, bank 0, bank 1, bank 2, and bank 3. Each bank has any number of rows, indicated by the rows in the table. Each row of each bank can store 4 macroblocks.

[0039] The video decoder 445 stores a macroblock row, e.g., macroblocks A0…A44, starting with bank 0 (A0…A3), and proceeding to bank 1 (A4…A7), bank 2 (A8…A11), bank 3 (A12…A15) and repeating until the last macroblock A44 is stored in bank 3. The next macroblock row, e.g., B0…B44, is stored starting from bank 2. The next macroblock row, e.g., C0…C44, is stored starting from bank 0, again. Each successive row is stored starting from either bank 0 or bank 2, in alternating fashion. For storing of a macroblock row from bank 0 and ending the macroblock row in bank 1, or starting storage of the macroblock row from bank 2, and ending the macroblock row in bank 3, a total of 16*N + 8 (where N is an integer, or the whole number of rows) macroblocks are needed for completely using all of the memory. However, an SDTV frame comprises 45 macroblocks across each row. To accommodate 45 macroblocks, the amount of memory required to store 56 macroblocks (N=3) is used.

[0040] As can be seen, a hole comprising the memory for storing 11 macroblocks is left empty (indicated by an X) per macroblock row. The amount of memory that actually stores data as percentage of the memory that is allocated for the macroblock row is approximately 80%, while approximately 20% is empty.

[0041] To reduce the proportion of unused memory, three SDTV frames can be concatenated horizontally. The concatenated frame can be stored as if the concatenated frame was a single frame.

[0042] Pursuant to the MPEG-2 standard, at least three frames are stored in the frame buffer at a given time. Two the frames are reference frames, a past prediction frame, and a future prediction frame. The third frame can be the most recently decoded frame, and can be decoded from the past prediction frame and the future prediction frame. Accordingly, the foregoing three frames can be stored in the frame buffer in the manner described above, as if they were a single horizontally concatenated frame.

[0043] Referring now to **FIGURE 7**, there is illustrated a block diagram of three horizontally concatenated frames. Each letter element represents a macroblock, while the subscript identifies the frame, and the following numeral represents the ordinal position of the macroblock within the macroblock row of the frame.

[0044] **FIGURE 8** is a block diagram of a frame buffer 452 storing the frames described in FIGURE 7. The video decoder 445 stores the first macroblock row of the first frame, $A_0$0…$A_0$44, starting with bank 0 ($A_0$0…$A_0$3), proceeding to bank 1 ($A_0$4…$A_0$7), to bank 2 ($A_0$8…$A_0$11), and to bank 3 ($A_0$12…$A_0$15), and repeating until the last macroblock $A_0$44 is stored in bank 3.

[0045] After storing the first macroblock row of the first frame, the video decoder 445 stores the first macroblock row of the second frame, $A_1$0… $A_1$44. The video decoder 445 starts with bank 3 ($A_1$0… $A_1$2), proceeding to bank zero ($A_1$3… $A_1$6), to bank 1 ($A_1$7…$A_1$10), and to bank 2

($A_1$11...$A_1$14) and repeating until the last macroblock $A_1$44 is stored in bank 2.

[0046] After storing the first macroblock row of the second frame the video decoder 445 stores the first macroblock row of the third frame, ($A_2$0...$A_2$44). The video recorder 445 starts with bank 2 ($A_2$0, $A_2$1), proceeding to bank 3 ($A_2$2...$A_2$5), to bank 0 ($A_2$6...$A_2$9) and to bank 1 ($A_2$10...$A_2$13), and repeating until the last macroblock $A_2$44 in row bank 1. After storing the first macroblock row of each frame, $A_0$0...$A_0$44, $A_1$0...$A_1$44, and $A_2$0...$A_2$44 the video decoder 445 skips the remaining portion of bank 1 and stores the second macroblock row of each frame $B_0$1...$B_0$44, $B_1$0...$B_1$44, and $B_2$0...$B_2$44 starting from bank 2. The next macroblock row of each frame, $C_0$0...$C_0$44, $C_1$0...$C_1$44, and $C_2$0...$C_2$44 starting from bank 0. Each successive row is stored starting from either bank 0 or bank 2 in an alternating fashion.

[0047] As can be seen, a hole comprising the memory for storing 1 macroblock is left empty (indicated by an X)per 3 macroblock rows. The amount of memory that actually stores data as a percentage of the memory that is allocated for the macroblock row is over 99%.

[0048] One embodiment of the present invention may be implemented as a board level product, as a single chip, application specific integrated circuit (ASIC), or with varying levels integrated on a single chip with other portions of the system as separate components. The degree of integration of the system will primarily be determined by speed and cost considerations. Because of the sophisticated nature of modern processors, it is possible to utilize a commercially available processor, which may be implemented external to an ASIC implementation of the present system. Alternatively, if the processor is available as an ASIC core or logic block, then the commercially available processor can be implemented as part of an ASIC device with various functions implemented as firmware.

[0049] While the invention has been described with reference to certain embodiments, it will be understood by those skilled in the art that various changes may be made and equivalents may be substituted without departing from the scope of the invention. In addition, many modifications may be made to adapt particular situation or material to the teachings of the invention without departing from its scope. Therefore, it is intended that the invention not be limited to the particular embodiment(s) disclosed, but that the invention will include all embodiments falling within the scope of the appended claims.